

**Content Reference Forum
Core Specification 1.0**

**CR10 001
Data Formats**

Revision: 1

Date: December 1, 2003

Status: Candidate Specification

Contents

1	INTRODUCTION.....	1
2	MODEL	1
2.1	ENTITIES.....	2
2.1.1	<i>Subject</i>	2
2.1.2	<i>Authority</i>	2
2.1.3	<i>Identification</i>	2
2.1.4	<i>Description</i>	2
2.1.5	<i>Value Chain</i>	3
2.1.6	<i>Target</i>	3
2.1.7	<i>Realization</i>	3
2.1.8	<i>Offer</i>	3
2.1.9	<i>Collection</i>	3
2.1.10	<i>Rendering Specifier</i>	3
2.2	RELATIONSHIPS	3
2.2.1	<i>R1</i>	3
2.2.2	<i>R2</i>	3
2.2.3	<i>R3</i>	4
2.2.4	<i>R4</i>	4
2.2.5	<i>R5</i>	4
2.2.6	<i>R6</i>	4
2.2.7	<i>R7</i>	5
2.2.8	<i>R8</i>	5
2.2.9	<i>R9</i>	5
3	CONTENT REFERENCE CORE DATA FORMATS.....	6
3.1	GENERAL MODEL MAPPING	6
3.1.1	<i>Reference</i>	6
3.1.2	<i>Target</i>	6
3.1.3	<i>Realization</i>	6
3.1.4	<i>Value Chain</i>	6
3.1.5	<i>Offer</i>	6
3.1.6	<i>Reference Collection</i>	6
3.1.7	<i>Rendering Specifier</i>	6
3.2	XML SCHEMA MAPPING.....	7
3.2.1	<i>XML Namespace Conventions</i>	7
3.2.2	<i>Structure</i>	7
3.2.3	<i>Scheme attributes</i>	7
3.2.4	<i>Libraries</i>	7
3.2.5	<i>Certification</i>	7
3.2.6	<i>Opaque data</i>	7
3.2.7	<i>Data types</i>	8
3.2.8	<i>Core substitutions</i>	8
3.3	CORE XML SCHEMA.....	8
3.3.1	<i>Global attribute groups</i>	8
3.3.1.1	<i>SCHEME</i>	8
3.3.2	<i>Core data types</i>	8
3.3.2.1	referenceType data type.....	8
3.3.2.2	offerType data type.....	8
3.3.2.3	targetType data type	9
3.3.2.4	valueChainType data type	9
3.3.2.5	realizationType	9
3.3.2.6	identifierType	10
3.3.2.7	referenceCollectionType.....	10
3.3.2.8	presentationType.....	10
3.3.3	<i>Core Data Elements</i>	10
3.3.3.1	reference element.....	10
3.3.3.2	offer element.....	10
3.3.3.3	target element.....	10
3.3.3.4	valueChain element.....	11
3.3.3.5	realization element.....	11
3.3.3.6	references element	11
3.3.3.7	presentation element	11
3.3.4	<i>Core Substitution Group Heads</i>	11
3.3.4.1	authority element	11
3.3.4.2	identification element	11
3.3.4.3	instance element.....	11
3.3.4.4	description element.....	11

3.3.4.5	participation element.....	11
3.3.4.6	offerPart element.....	11
3.3.4.7	targePart element	11
3.3.4.8	renderingSpecifier element.....	12
3.3.5	<i>Utility Elements</i>	12
3.3.5.1	itemref element	12
3.3.5.2	opaqueElement element	12
3.3.5.3	certification element	12
3.3.6	<i>Library Elements</i>	12
3.3.6.1	authorityLibrary element	12
3.3.6.2	participationLibrary element.....	12
3.3.7	<i>Core substitution elements</i>	13
3.3.7.1	server element.....	13
3.3.7.2	identifier element	13
3.3.7.3	location element.....	13
3.3.7.4	bObject element	13
3.3.7.5	consumer element	13
3.3.7.6	proposition element.....	13
3.3.7.7	contribution element	14
3.3.7.8	title element	14
3.3.7.9	hidden element.....	14
APPENDIX A -	SCHEMA	15
APPENDIX B -	REFERENCES.....	20

1 Introduction

The vision for CR Forum (CRF) is to define a framework that enables transparent and dynamic acquisition and sharing of multimedia content across a wide range of networks and devices used by different communities while ensuring compliance of the appropriate commercial terms for a given consumer and value chain context. A key concept of the CRF Architecture is that the information required to enable this can be captured as a “Content Reference” and processed by specialized resolution services. Content Reference is a data construct containing reference and other metadata associated with multimedia content. It enables access to an adapted version and format of the multimedia content suited for and requested by a particular consumer. Content Reference data package can capture such information as:

- what is the identification and description of the content in question,
- what is the commercial environment of the consumer, e.g., country, language, affiliations, preferred payment methods, etc.
- what is the technical environment of the consumer, e.g., rendering capabilities and preferences, content protection methods, content delivery options, etc.
- what is the value chain (participants and the nature of their participation) of the content in question,
- how to resolve the content reference.

The CR Architecture (CRA) basic premise is that, in order to provide users with the right content under the right commercial terms, we need to have:

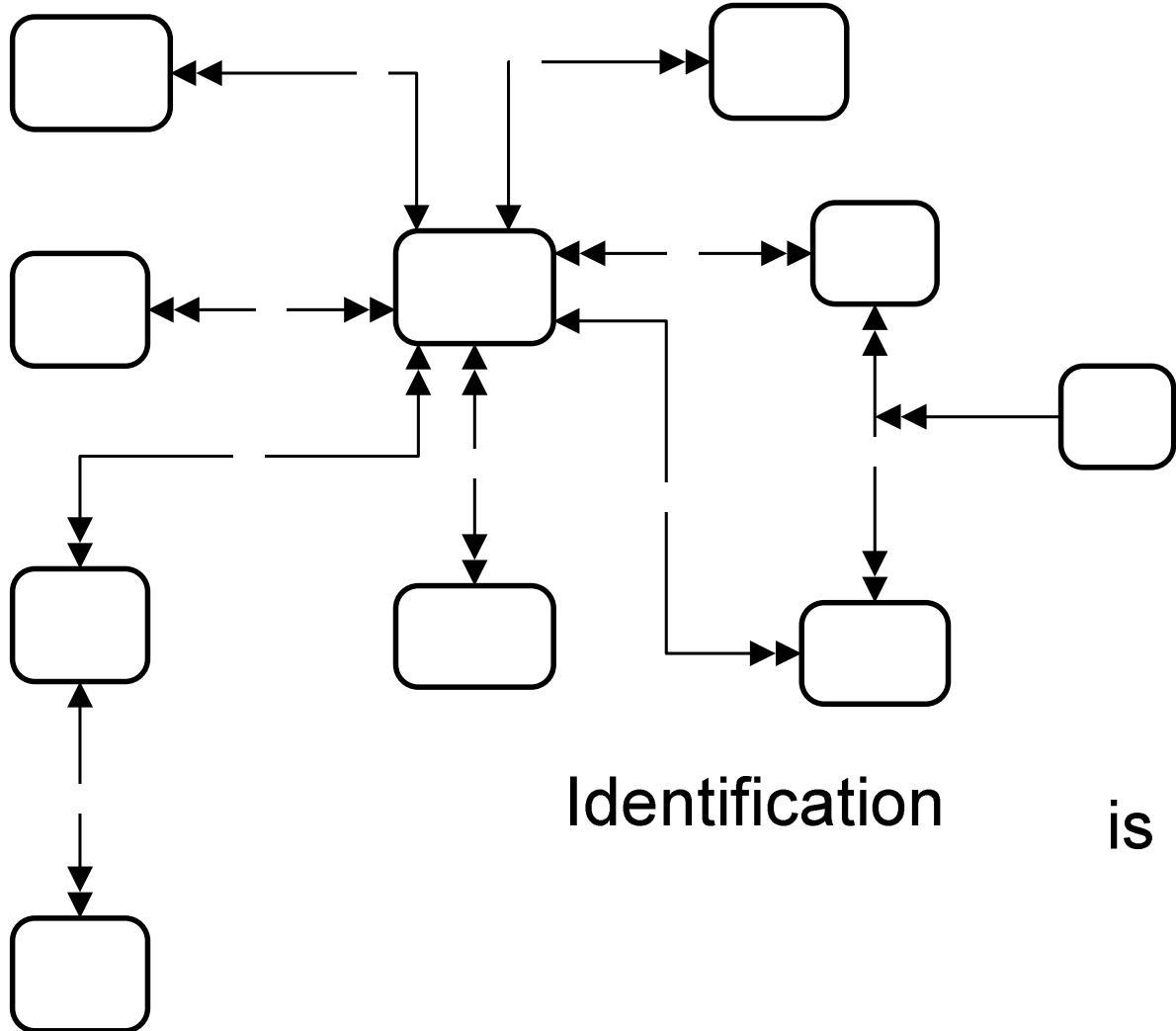
- client-side elements to collect information which is relevant to determining the right content and the right terms and to incorporate this information into content references,
- server-side elements to determine the right content and the right terms by processing the content reference against the applicable business rules (e.g. electronic contracts, policies, constraints), and to generate an offer for the client-side (or to perform other contractually specified actions) in response to the client request. We call such elements *Reference Services*,
- client-side elements to enable users to act on the server-side response (e.g., an offer).

This document describes data model and code data formats of Content References. This is a core specification that can be extended and/or constrained in *profiles* defined for selected usage domains.

2 Model

The Content Reference Core Data Formats (CRCFs) are based on the Content Reference Data Model (CRDM). The CRDM is a relational data model described by the following simplified

ERD:



Note that, to avoid unnecessary complication for this discussion, certain formalizing entities (for many-to-many relationships) are not shown in the above diagram. Further, this model is situated at a high abstraction level. In practice, a large number of additional entities and relationships will exist.

2.1 Entities

2.1.1 Subject

At the heart of the CRDM is the Content Reference *subject*. This represents the item of interest in a Content Reference. In a Content Reference system, subjects typically represent abstract content without regard to content encoding, delivery, or protection.

2.1.2 Authority

An *authority* entity identifies a content reference system and describes the *Reference Service* (RS) bindings of a number of subjects. An authority provides context to CR *nodes* and prescribes the methods by which subjects can be *resolved*.

2.1.3 Identification

A subject can be known by any number of identifiers. An *identification* entity associates an identifier—and attendant identifier space—with a CR subject. The identifiers encapsulated in identification entities are independent of a subject entity's primary key in any particular database instance.

2.1.4 Description

A *description* entity provides human-oriented information describing a subject. A CR node that has user interface features uses description information to enable user interaction with a subject and to enhance user experience.

2.1.5 Value Chain

A *value chain* entity represents a portion of a *Value Chain* in a particular commercial context. A Value Chain identifies and describes *participants* that have contributed to the value of subjects and the nature of their *participation*.

2.1.6 Target

A *target* entity encapsulates the commercial and technical environment of a *consumer*. Target information is used by Reference Services to compute *offers* and *directions* aimed at a consumer's individual circumstances.

2.1.7 Realization

A *realization* entity encapsulates the data necessary to use a subject. The range of information in a realization includes data formats and protection details for digital content, delivery parameters, quality of service data, and the subject realization data itself.

2.1.8 Offer

An *offer* entity represents an invitation to consumption of subjects. It describes the necessary conditions for execution and revocation along with any constraints on consumption. An offer is modelled as a relationship formalizer binding realization instance entities to target entity instances.

2.1.9 Collection

A *collection* entity is an arbitrary collection of subject instances. It represents a grouping of subjects for an application-defined purpose.

2.1.10 Rendering Specifier

A *rendering specifier* entity specifies how subjects in a collection instance should be rendered as a group.

2.2 Relationships

2.2.1 R1

Endpoint	Cardinality	Conditionality	Annotation
subject	many	conditional	An authority may manage subjects .
authority	many	unconditional	A subject is managed by authorities .

Relationship R1 establishes associations between authorities and subjects. Since a Content Reference system is concerned with a number of subjects, R1 has cardinality “many” on the subject side. The relationship is conditional on the subject side to account for the situation where an authority is established that has not been populated with subject entities.

Complex interoperability situations between content reference networks might result in a subject that is known to multiple, distinct systems and that must have its multiple authority relationships captured. Thus the cardinality of R1 on the authority side is “many.” R1 is unconditional on the authority side since subjects are always established in the context of some Content Reference network(s).

For simplicity, the formalizer for this relationship is not presented here.

2.2.2 R2

Endpoint	Cardinality	Conditionality	Annotation
subject	one	unconditional	An identification identifies a subject .
identification	many	unconditional	A subject is known by identifications .

Relationship R2 associates a subject with the identifiers by which it is known. An identifier refers to exactly one subject so that R2 is unconditional with cardinality “one” on the subject side.

A subject can be identified in any number of identifier spaces so the cardinality of R2 is “many” on the identification side. R2 is unconditional on the identification side so that each subject is known in at least one identifier space.

2.2.3 R3

Endpoint	Cardinality	Conditionality	Annotation
subject	one	conditional	A description describes a subject .
description	many	conditional	A subject may be described by descriptions .

Relationship R3 associates a subject with its descriptive information. Each description corresponds to exactly one subject so that R3 is unconditional with cardinality “one” on the subject side.

A subject can be described under any number of descriptive systems so the cardinality of R3 is “many” on the description side. A subject is not required to have associated descriptive data so R3 is conditional on the description side.

2.2.4 R4

Endpoint	Cardinality	Conditionality	Annotation
subject	many	conditional	A value chain may be a commercial context for subjects .
value chain	many	conditional	A subject may have commercial contexts of value chains .

Relationship R4 establishes associations between subjects and value chains. Since a subject can be distributed simultaneously in multiple commercial contexts, the cardinality of R4 is “many” on the value chain side. To accommodate situations where a subject is created but has not yet been placed into any commercial context, R4 is conditional on the value chain side.

Because a value chain entity represents a portion of a Value Chain, it can be associated with more than one subject. Thus R4 has cardinality “many” on the subject side. To accommodate situations where a value chain entity is created in anticipation of association with subjects, R4 is conditional on the subject side.

For simplicity, the formalizer for this relationship is not presented here.

2.2.5 R5

Endpoint	Cardinality	Conditionality	Annotation
subject	many	conditional	A target may request subjects .
target	many	conditional	A subject may be requested by targets .

Relationship R5 associates target entities with subject entities for the purpose of resolution requests. Instances of R5 exist for the duration of an RS invocation.

R5 has cardinality “many” on the subject side because a consumer can simultaneously request resolution for any number of subjects. It is conditional on the subject side since a target environment exists even when it is not involved in a request.

Since a resolution request can be issued by any number of consumers simultaneously for the same subject, R5 has cardinality “many” on the target side. A subject, at any time, can be associated with no requests so R5 is conditional on the target side.

For simplicity, the formalizer for this relationship is not presented here.

2.2.6 R6

Endpoint	Cardinality	Conditionality	Annotation
target	many	conditional	A realization may be offered to targets .
realization	many	conditional	A target may be offered realizations .

Relationship R6 associates realizations with targets. The relationship is formalized by offer entities that capture the fulfillment conditions for each relationship instance. An offer can pertain to multiple targets so the relationship has a cardinality of “many” on the target side. R6 is conditional on the target side since a realization can exist without any offers.

The relationship has cardinality “many” on the realization side since, at a given time, a target may have offers for more than one realization pending. R6 is conditional on the realization side since targets exist independent of any offered subject realizations.

Additionally, R6 itself has cardinality “many.” This captures cases where multiple offers are pending for the same target and realization pair (e.g. different pricing and licensing options).

2.2.7 R7

Endpoint	Cardinality	Conditionality	Annotation
subject	one	unconditional	A realization projects a subject .
realization	many	conditional	A subject may be projected by realizations .

Relationship R7 associates realizations with their subjects. On the subject side, the relationship is unconditional with cardinality “one” to model the fact that a realization corresponds to exactly one subject.

The relationship has cardinality “many” on the realization side since a particular subject can be expressed in any number of concrete forms. Since a subject can, during its lifecycle, have zero realizations, R7 is conditional on the realization side.

2.2.8 R8

Endpoint	Cardinality	Conditionality	Annotation
subject	many	conditional	A collection may group subjects .
collection	many	conditional	A subject may be grouped in collections .

Relationship R8 groups subjects into collections. On the subject side, the relationship is conditional with cardinality “many” to model the collecting of subjects into, possibly empty, sets.

The relationship has cardinality “many” on the collection side because the model allows subjects to be grouped into any number of collections. It is conditional because any particular subject need not be member of any collections.

For simplicity, the required formalization entity is not shown.

2.2.9 R9

Endpoint	Cardinality	Conditionality	Annotation
collection	one	unconditional	A rendering specifier directs rendering of a collection .
rendering specifier	many	conditional	A collection may have rendering directed by rendering specifiers .

Relationship R9 associates rendering specifiers with their collections. On the collection side, the relationship is unconditional with cardinality “one.” That is, a rendering specifier directs the presentation of precisely one collection.

Any number of rendering specifiers can be applied to a collection thus the relationship is unconditional with cardinality “many” on the rendering specifier side.

3 Content Reference Core Data Formats

3.1 General Model Mapping

The CRDM provides a conceptual data framework for CR systems. At a particular moment, the state of a CR system (or systems) is defined by the set of CRDM entity and relationship instances in existence at that moment.

In a given CR system, the entity and relationship instances will exist on a number of CR system nodes. Each node may realize only a portion of the state of the entire system and multiple nodes may realize identical information. *Content Reference Data Packages* (CRDPs), then, are intended to represent a part of the state of a CR system for the purpose of communicating knowledge among nodes.

The mapping approach taken in this specification defines low-level elements corresponding to entities of the CRDM as well as composite structures that express relationships. The use of fine-grained entity mapping allows CRDF components to be used in multiple contexts.

CRDFs define the syntax and semantics of CRDPs. There are five core CRDFs:

3.1.1 Reference

A *reference* (or *Content Reference*) is a projection of the subject, authority, identification, description, and value chain entities along with relationships R1, R2, R3, R4 with respect to a single subject entity instance. The structure utilizes the value chain structure (see 2.1.4). Taken alone, a reference structure declares the identity, description, and a value chain projection of a subject. Thus the reference structure is the primary means of communicating concrete content information between consumer-oriented Content Reference Network nodes.

3.1.2 Target

A *target* is a projection of the target entity for a single instance. The structure is used to communicate environmental information between Content Reference Network nodes. In particular, a Target structure is used to inform a Reference Service or other network service of the capabilities, constraints, circumstances, and preferences of a consumer.

3.1.3 Realization

A *realization* is a projection of the realization entity for a single instance. A realization structure is used to communicate details of concrete subject instances between Content Reference Network nodes. In particular, it is used to send consumption details to a consumption-oriented node.

3.1.4 Value Chain

A *value chain* is a projection of the CRDM value chain entity. The structure contains a commercial context in the form of participation entries. Each participation entry denotes a claim of value for one or more *value-chain participants*.

3.1.5 Offer

An *offer* is a projection of the offer entity along with a partial projection of and instance of relationship R6 with respect to a set of offer instances and realizations and a single target. An offer is typically sent to a consumption-oriented node associated with a particular target.

3.1.6 Reference Collection

A *reference collection* is a projection of the collection entity and relationship R8 with respect to a single collection entity instance.

3.1.7 Rendering Specifier

A *rendering specifier* is a projection of the rendering specifier entity and relationship R9 with respect to a single rendering specifier instance.

3.2 XML Schema Mapping

CRDPs are encoded using XML. The corresponding CRDFs are expressed using XML Schema. XML encoding enables CRDPs to be embedded into modern network communication mechanisms such as SOAP.

3.2.1 XML Namespace Conventions

For convenience, namespace prefixes are assigned to XML namespaces for reference within this document alone. Implementations are not required to use the prefixes listed here.

Prefix	Namespace
cr	http://www.crforum.org/namespace/crdf-core
xs	http://www.w3.org/2001/XMLSchema
xsi	http://www.w3.org/2001/XMLSchema-instance

3.2.2 Structure

The core schema, CRDF-CORE, defines the structure of the five CRDFs by composing XML Schema abstract elements. The abstract elements serve as head elements for substitution groups used in extension schemas and in the core schema itself.

3.2.3 Scheme attributes

Certain elements of the schema specify XML Schema simple content. However, the semantics of content in some of those elements are undefined in the core schema. In such cases, the elements accept attributes that select the semantics of the element content when it is not implied by a particular system.

For an element, its *scheme* attribute, if present, is a URI that encapsulates the interpretation context of its content. A *schemeref*, if present in an element, denotes an interpretation context as a scheme does but is a reference to a declared namespace prefix visible to the element rather than a UR. A schemeref must be interpreted exactly as if a scheme attribute was present containing the namespace value associated with the schemeref reference.

If both a schemeref and scheme are present in an element, the scheme element takes precedence.

In the core schema, the acceptable values of scheme attributes are unconstrained—other than that they must be legal XML namespace values—and optional. Extensions to the core schema should define conditionality and value constraints as well as the semantics of particular values.

3.2.4 Libraries

Library structures are provided in the schema to model relationships with cardinality “many” and to aid in the unbundling of elements for various network services. The use of library elements is optional for any CRDP. If not used, exact duplicates of entity instances must be used instead to model “many.”

3.2.5 Certification

The XML encoding provides for attributed assertions in the form of certification substitution elements. These are used within CRDPs themselves when fact assertion must endure beyond a particular event or transaction. Otherwise, facts (such as message validity) can be asserted by any means suitable to a particular application or communication channel.

Where used in this specification, specific constraints and semantics are defined for the certification context.

3.2.6 Opaque data

The schema allows some elements to be replaced by elements not generally interpretable. Each such opaque element must, when interpreted by an authorized node, correspond to a CRDP element that is legal in that context.

3.2.7 Data types

For the five CRDP types, the core schema uses XML data types for their roots. This allows other data formats to include CR data by typing elements rather than referring to global elements. The schema also defines five concrete elements that correspond to the data types so that valid CRDPs can be instantiated as standalone XML elements and document roots using the core schema alone.

3.2.8 Core substitutions

The core schema defines several elements that can be directly substituted for structural elements. These are defined in order to allow systems to be based on the core schema alone.

3.3 Core XML Schema

3.3.1 Global attribute groups

3.3.1.1 SCHEME

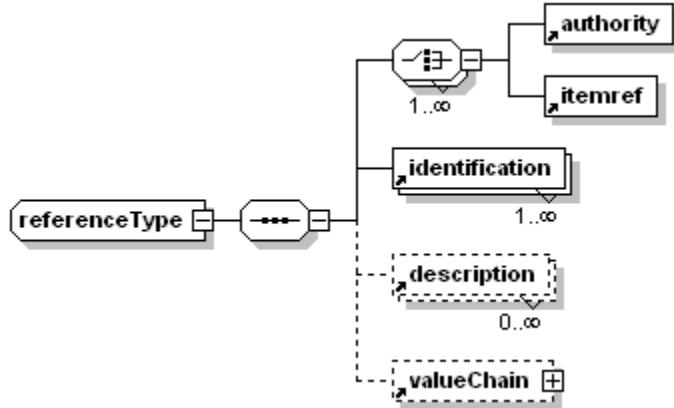
The SCHEME attribute group defines the scheme and schemeref attributes. The scheme attribute is of type xs:anyURI and the schemeref attribute is of type xs:NCName.

3.3.2 Core data types

The core XML data types define the top-level structures of CRDPs.

3.3.2.1 *referenceType* data type

The *referenceType* data type defines the structure of a reference CRDP. A reference corresponds to exactly one subject entity instance and a subset of its primary relationships. Its top-level form is:



It contains 4 main sections: authority, identification, description, and value chain.

The authority section is a collection of authority elements (see section 2.3.4.1) or references to authority elements from a library (see sections 2.3.6.1 and 2.3.5.1). Any combination of authority elements and itemref elements is permitted.

The identification section is a collection of identification elements (see section 2.3.4.2) by which a single reference subject is known.

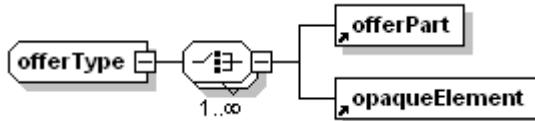
The description section is an optional list of descriptive elements (see section 2.3.4.4) corresponding to the reference subject.

The valueChain section is an optional list of value chain declarations (see section 2.3.2.4) corresponding to the reference subject.

The type has an optional attribute, id, of type xs:ID which is used to refer to elements of the type within a document.

3.3.2.2 *offerType* data type

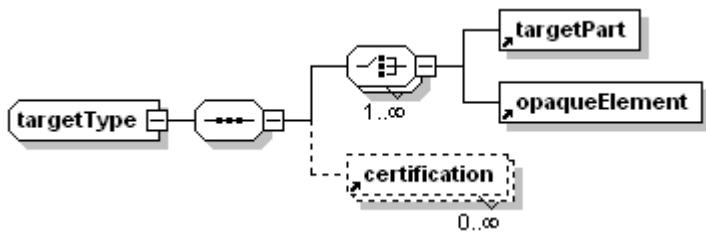
The offerType data type defines the structure of an offer CRDP. Its form is:



An element of this type contains a sequence of offer parts (see section 2.3.4.6) or their opaque counterparts (see section 2.3.5.2). An element of this type may contain an automatic attribute of type xs:boolean. If not present, its value is presumed false. If automatic is true, it indicates that the Offer entity instance described by the element is to be automatically consumed upon receipt of the associated offer CRDP.

3.3.2.3 targetType data type

The targetType data type defines the structure of a target CRDP. Its form is:

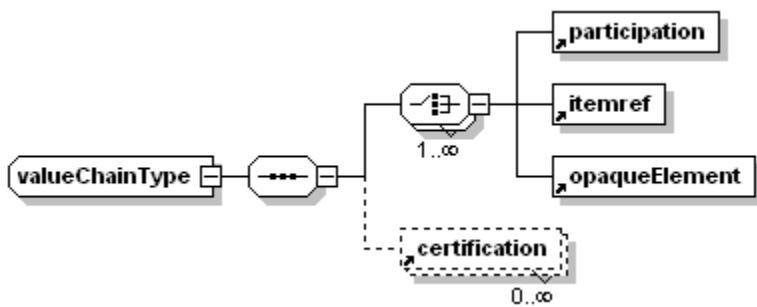


An element of type targetType contains a sequence of target parts (see section 2.3.4.7) or their opaque counterparts (see section 2.3.5.2) followed by optional certifications (see section 2.3.5.3).

Each certification element must attest to the authenticity of one or more target parts (or opaque target counterparts) and the operational context in which the parts are certified. A certification and the elements it certifies must reside in the same targetType structure. How the context is declared depends on the certification mechanism used and the type of message that contains the targetType structure. In the case of an opaque element, the application may choose to certify the either opaque or plain version of the element as suits its requirements.

3.3.2.4 valueChainType data type

The valueChainType data type describes the structure of a value chain CRDP. Its form is:



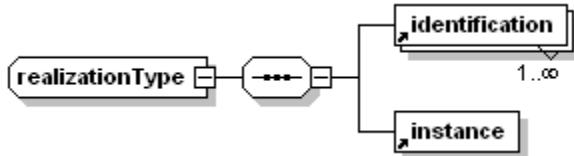
An element of type valueChainType contains a sequence of participation elements (see section 2.3.4.5), library references (see sections 2.3.5.1 and 2.3.6.2), or opaque participation counterparts (see section 2.3.5.2) followed by an optional list of certifications (see section 2.3.5.3).

Participation elements, item references, and opaque representations of participations elements can be intermixed in an element of valueChainType type and their respective order carries no semantics.

Each certification element must attest to the authenticity of one or more participation elements (or their opaque counterparts), an identification element, and the operational context in which the parts are certified. How context is declared depends on the certification method used and the type of message that contains the valueChainType structure. If any of the certified elements is an itemref element, certification must be verifiable with respect to the actual element from the participationLibrary (see section 2.3.6.2). In the case of an opaque element, the application may choose to certify either the opaque or plain version of the element as suits its requirements.

3.3.2.5 realizationType

The realizationType data type describes the structure of a realization CRDP. Its form is:



An element of type `realizationType` contains a sequence of identifiers (see section 2.3.4.2) followed by a subject instance (see section 2.3.4.3). The identifier sequence declares a set of identifiers by which the contained instance is known.

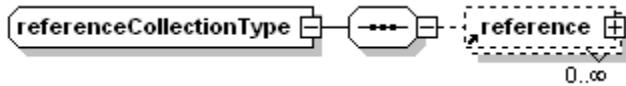
3.3.2.6 *identifierType*

The `identifierType` data type identifies abstract or packaged content. It has a complex content model consisting of simple content and optional attributes. The content is of type `xs:normalizedString`.

The optional `SCHEME` attribute group (see section 2.3.1.1) selects an identification scheme or namespace if the implementation supports more than one.

3.3.2.7 *referenceCollectionType*

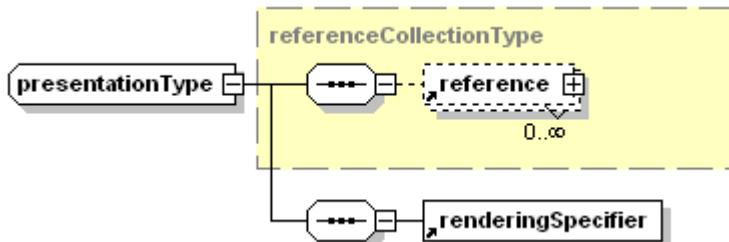
The `referenceCollectionType` data type describes a collection of subjects. Its form is:



An element of type `referenceCollectionType` contains a sequence of reference elements (see section 2.3.3.1). Any number, including zero, of reference elements can be included in the sequence to capture the cardinality and conditionality requirements of the CRDM (see section 1.2.8).

3.3.2.8 *presentationType*

The `presentationType` data type represents a rendering specifier and its corresponding collection. It has the form:



The `presentationType` data type is a derivation, by extension, of the `referenceCollectionType` data type. The structure consists of a collection of reference elements (see section 2.3.3.1) followed by a rendering specifier (see section 2.3.4.8).

3.3.3 Core Data Elements

3.3.3.1 *reference element*

A reference element is a global element of type `referenceType` (see section 2.3.2.1). It can be used to create standalone reference CRDPs by making it the root element of an XML document.

3.3.3.2 *offer element*

An offer element is a global element of type `offerType` (see section 2.3.2.2). It can be used to create standalone offer CRDPs by making it the root element of an XML document.

3.3.3.3 *target element*

A target element is a global element of type `targetType` (see section 2.3.2.3). It can be used to create standalone target CRDPs by making it the root element of an XML document.

3.3.3.4 *valueChain element*

A valueChain element is a global element of type valueChainType (see section 2.3.2.4). It can be used to create standalone CRDPs by making it the root element of an XML document.

3.3.3.5 *realization element*

A realization element is a global element of type realizationType (see section 2.3.2.5). It can be used to create standalone CRDPs by making it the root element of an XML document.

3.3.3.6 *references element*

A references element is a global element of type referenceCollectionType (see section 2.3.2.7). It can be used to create standalone content reference collections by making it the root of an XML document.

3.3.3.7 *presentation element*

A presentation element is a global element of type presentationType (see section 2.3.2.8). It can be used to create a standalone presentation by making it the root of an XML document.

3.3.4 Core Substitution Group Heads

3.3.4.1 *authority element*

An authority element is an abstract element that forms the head of the authority substitution group. Authority elements are instances of the authority entity in the CRDM.

3.3.4.2 *identification element*

An identification element is an abstract element that forms the head of the identification substitution group. Identification elements are instances of the identification entity in the CRDM.

3.3.4.3 *instance element*

An instance element is an abstract element that forms the head of the instance substitution group. Instance elements describe the technical details of a realization entity in the CRDM and are a component of the realizationType data type (see section 2.3.2.5).

3.3.4.4 *description element*

A description element is an abstract element that forms the head of the description substitution group. Description elements are instances of the description entity in the CRDM.

3.3.4.5 *participation element*

A participation element is an abstract element that forms the head of the participation substitution group. Participation elements are detailed entries of the value chain entity in the CRDM.

3.3.4.6 *offerPart element*

An offerPart element is an abstract element that forms the head of the offerPart substitution group. An offerPart element corresponds to an offer entity instance in the CRDM and is a component of the offerType datatype (see section 2.3.2.2).

3.3.4.7 *targetPart element*

A targetPart element is an abstract element that forms the head of the targetPart substitution group. A targetPart element corresponds to part of a target entity instance in the CRDM and is a component of the targetType datatype (see section 2.3.2.3).

3.3.4.8 *renderingSpecifier element*

A renderingSpecifier element is an abstract element that forms the head of the renderingSpecifier substitution group. A renderingSpecifier element is an XML projection of the rendering specifier CRDM entity (see section 2.1.7).

3.3.5 Utility Elements

3.3.5.1 *itemref element*

An itemref element has an empty content model and one attribute, *idref* of type xs:IDREF. The idref attribute is used to select an item from a library (see section 2.3.6). An itemref can be used in place of either a participation element or an authority element. In each case, an itemref must refer to an item in an associated library that contains a participation element or authority element that would be legal at that point in the CRDP.

3.3.5.2 *opaqueElement element*

The opaqueElement is an abstract XML Schema element that forms the head of the opaqueElement substitution group.

An opaqueElement is a stand-in for another CRDP element and is not generally interpretable by Content Reference nodes. A node can be specifically programmed to interpret certain opaqueElement forms.

OpaqueElements are included in the specification to support information hiding mechanisms such as encryption and out-of-band resolution for privacy purposes.

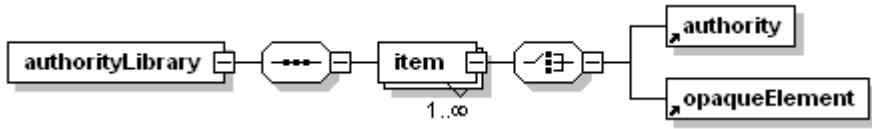
3.3.5.3 *certification element*

The certification element is an abstract XML Schema element that forms the head of the certification substitution group. An element that substitutes for a certification element asserts some claim of fact with respect to the enclosing CRDP.

3.3.6 Library Elements

3.3.6.1 *authorityLibrary element*

An authorityLibrary element is a collection of authority elements. It has the form:

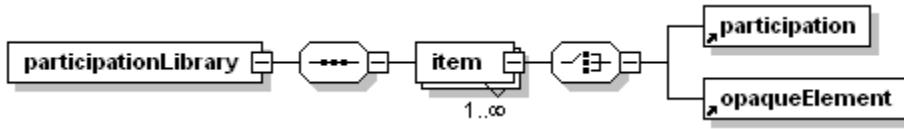


The structure of an authorityLibrary element consists of a sequence of item elements. An item element has a complex content model consisting of either an authority element (see section 2.3.4.1) or the opaque counterpart (see section 2.3.5.2) of an authority element. An item element has a required *id* attribute of type xs:ID. This attribute may be referred to in any number of itemref elements (see section 2.3.5.1).

The containment of authority elements in a library implies no semantics and the library's elements are unordered.

3.3.6.2 *participationLibrary element*

A participationLibrary element is a collection of authority elements. It has the form:



The structure of a participationLibrary element consists of a sequence of item elements. An item element has a complex content model consisting of either a participation element (see section 2.3.4.5) or the opaque counterpart (see section 2.3.5.2) of an authority element. An item element

has a required *id* attribute of type xs:ID. This attribute may be referred to in any number of itemref elements (see section 2.3.5.1).

The containment of participation elements in a library implies no semantics and the library's elements are unordered.

3.3.7 Core substitution elements

The core XML schema defines several elements that can be directly substituted for corresponding abstract structural elements. The certification substitution group does not have a concrete substition in this specification.

3.3.7.1 *server element*

The server element type belongs to the authority substitution group (see section 2.3.4.1). It has a complex content model with a content type of xs:anyURI and one attribute group. The data in a server element must be a URL that locates a capable reference service on the Internet.

The optional SCHEME attribute group (see section 2.3.1.1) selects a communication and processing profile if the implementation supports more than one.

3.3.7.2 *identifier element*

The identifier element type belongs to the identification substitution group (see section 2.3.4.2). It has a complex content model of type identifierType (see section 2.3.2.6).

3.3.7.3 *location element*

The location element type belongs to the instance substitution group (see section 2.3.4.3). It has a complex content model with a content type of xs:anyURI and one attribute group. The content in a location element must be a URL that locates a consumption-oriented subject package on the public Internet.

The optional SCHEME attribute group (see section 2.3.1.1) selects an interpretation of the remote data if the implementation supports more than one package type.

3.3.7.4 *bObject element*

The bObject element type belongs to the instance substitution group (See section 2.3.4.3). It has a complex content model with a content type of xs:base64binary and one attribute group. The content in a bObject element must be a consumption-oriented subject package encoded as Base64 text.

The optional SCHEME attribute group (see section 2.3.1.1) selects an interpretation of the packaged data if the implementation supports more than one package type.

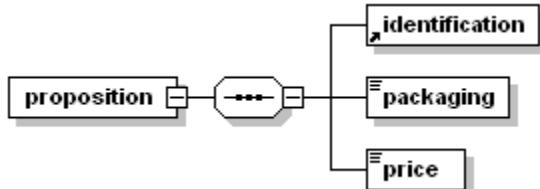
3.3.7.5 *consumer element*

The consumer element type belongs to the targetPart substitution group (see section 2.3.4.7) and represents a consumer's consumption environment. The type has a complex content model with simple content of type xs:normalizedString and one attribute group. A consumer element contains a value that identifies a consumption environment or consumer in some registry.

The optional SCHEME attribute group (see section 2.3.1.1) selects the consumption environment registry for those implementations that declare more than one.

3.3.7.6 *proposition element*

The proposition element type belongs to the offerPart substitution group (see section 2.3.4.6) and represents a simple offer for subject consumption. Its form is:



A proposition element has a complex content model consisting of a sequence of three elements: identifier, packaging, and price.

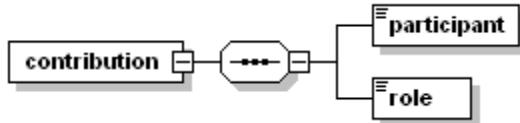
An identifier element in a proposition context is a reference to the same element type that is used in the identification substitution group. It refers to a subject via an indirect relation with one of the subject's identifiers. Note that this relationship is not part of the core CFDM but is implied by the composition of R2 and R6.

A packaging element has a complex content model consisting of xs:normalizedString data and one attribute. The content data contains the identifier of some packaging configuration with respect to a particular identification scheme. The implementation may load the semantics of the identifier with any number of packaging and constraint characteristics (e.g. CODEC, QoS, and license duration). The optional SCHEME attribute group selects an identification scheme if more than one is known to an implantation.

A price element has a complex content model consisting of xs:integer content and one attribute. The content data contains the integer-scaled price at which the corresponding subject and packaging are being offered. If an implementation works with multiple currencies, the optional SCHEME attribute group can be used to select the evaluation mode of the element's content.

3.3.7.7 *contribution element*

The contribution element type belongs to the participation substitution group (see section 2.3.4.5) and represents a simple relational model of value chain participation. Its form is:



A contribution element has a complex content model consisting of a sequence of two elements: participant and role.

A participant element has a complex content model with content consisting of xs:normalizedString data and one attribute. The content data contains the identifier of a value chain participant with respect to some identification space or registry. The optional SCHEME attribute group (see section 2.3.1.1) selects a space or registry if more than one is known to the implementation.

A role element has a complex content model with content consisting of xs:normalizedString data and one attribute. The content data contains the identifier of a value chain participation role with respect to some identification mechanism. The optional SCHEME attribute group (see section 2.3.1.1) selects an identification scheme if more than one is known to the implementation.

3.3.7.8 *title element*

The title element type belongs to the description substitution group (see section 2.3.4.4). It has a complex content model with a content type of xs:normalizedString and one attribute. The data in a title element describes the subject by its human-known name.

The optional language attribute, of type xs:language, selects the human language of the element content and allows applications to select appropriate descriptions to present to users.

3.3.7.9 *hidden element*

The hidden element type belongs to the opaqueElement substitution group (see section 2.3.5.2). It has a complex content model consisting of xs:base64Binary data and one attribute. Other than its encoding in base64, the content of a hidden element is unspecified by the CRDF.

If an implementation needs to distinguish between interpretation modes of hidden elements, it may use the SCHEME attribute group (see section 2.3.1.1) for that purpose.

Appendix A - Schema

```

<?xml version="1.0" encoding="UTF-8"?>
<xs:schema targetNamespace="http://www.crforg.org/namespace/crdf-core"
  xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns="http://www.crforg.org/namespace/crdf-core"
  elementFormDefault="qualified" attributeFormDefault="unqualified">
  <xs:attributeGroup name="SCHEME">
    <xs:attribute name="scheme" type="xs:anyURI" use="optional"/>
    <xs:attribute name="schemeref" type="xs:NCName" use="optional"/>
  </xs:attributeGroup>
  <!--Core Data Types-->
  <xs:complexType name="referenceType">
    <xs:annotation>
      <xs:documentation>content reference data structure</xs:documentation>
    </xs:annotation>
    <xs:sequence>
      <xs:choice maxOccurs="unbounded">
        <xs:element ref="authority"/>
        <xs:element ref="itemref"/>
      </xs:choice>
      <xs:element ref="identification" maxOccurs="unbounded"/>
      <xs:element ref="description" minOccurs="0" maxOccurs="unbounded"/>
      <xs:element ref="valueChain" minOccurs="0"/>
    </xs:sequence>
    <xs:attribute name="id" type="xs:ID" use="optional"/>
  </xs:complexType>
  <xs:complexType name="offerType">
    <xs:annotation>
      <xs:documentation>offer data structure</xs:documentation>
    </xs:annotation>
    <xs:choice maxOccurs="unbounded">
      <xs:element ref="offerPart"/>
      <xs:element ref="opaqueElement"/>
    </xs:choice>
    <xs:attribute name="automatic" type="xs:boolean" use="optional" default="false"/>
  </xs:complexType>
  <xs:complexType name="targetType">
    <xs:annotation>
      <xs:documentation>consumption environment data structure</xs:documentation>
    </xs:annotation>
    <xs:sequence>
      <xs:choice maxOccurs="unbounded">
        <xs:element ref="targetPart"/>
        <xs:element ref="opaqueElement"/>
      </xs:choice>
      <xs:element ref="certification" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="valueChainType">
    <xs:annotation>
      <xs:documentation>container for value chain entries</xs:documentation>
    </xs:annotation>
    <xs:sequence>
      <xs:choice maxOccurs="unbounded">
        <xs:element ref="participation"/>
        <xs:element ref="itemref"/>
        <xs:element ref="opaqueElement"/>
      </xs:choice>
      <xs:element ref="certification" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="realizationType">
    <xs:annotation>
      <xs:documentation>realized subject data structure</xs:documentation>
    </xs:annotation>
    <xs:sequence>
      <xs:element ref="identification" maxOccurs="unbounded"/>
      <xs:element ref="instance"/>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="identifierType">
    <xs:annotation>
      <xs:documentation>content identifier data type</xs:documentation>
    </xs:annotation>
    <xs:simpleContent>
      <xs:extension base="xs:normalizedString">
        <xs:attributeGroup ref="SCHEME"/>
      </xs:extension>
    </xs:simpleContent>
  </xs:complexType>
  <xs:complexType name="referenceCollectionType">
    <xs:annotation>
      <xs:documentation>collection of content references</xs:documentation>
    </xs:annotation>
    <xs:sequence>
      <xs:element ref="reference" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>

```

```
<xs:complexType name="presentationType">
  <xs:annotation>
    <xs:documentation>media vector with rendering hints</xs:documentation>
  </xs:annotation>
  <xs:complexContent>
    <xs:extension base="referenceCollectionType">
      <xs:sequence>
        <xs:element ref="renderingSpecifier"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
<!--Concreted Core Data Types-->
<xs:element name="reference" type="referenceType">
  <xs:annotation>
    <xs:documentation>concrete content reference</xs:documentation>
  </xs:annotation>
</xs:element>
<xs:element name="offer" type="offerType">
  <xs:annotation>
    <xs:documentation>an offer or collection of offers</xs:documentation>
  </xs:annotation>
</xs:element>
<xs:element name="target" type="targetType">
  <xs:annotation>
    <xs:documentation>describes consumption target</xs:documentation>
  </xs:annotation>
</xs:element>
<xs:element name="valueChain" type="valueChainType">
  <xs:annotation>
    <xs:documentation>correlated collection of participation elements</xs:documentation>
  </xs:annotation>
</xs:element>
<xs:element name="realization" type="realizationType">
  <xs:annotation>
    <xs:documentation>realized subject</xs:documentation>
  </xs:annotation>
</xs:element>
<xs:element name="references" type="referenceCollectionType">
  <xs:annotation>
    <xs:documentation>concreted referenceCollectionType</xs:documentation>
  </xs:annotation>
</xs:element>
<xs:element name="presentation" type="presentationType">
  <xs:annotation>
    <xs:documentation>concreted playlistType</xs:documentation>
  </xs:annotation>
</xs:element>
<!--Core Substitution Group Heads-->
<xs:element name="authority" abstract="true">
  <xs:annotation>
    <xs:documentation>description/location/identification of reference services capable/authorized to operator on this reference</xs:documentation>
  </xs:annotation>
</xs:element>
<xs:element name="identification" abstract="true">
  <xs:annotation>
    <xs:documentation>identity of reference subject</xs:documentation>
  </xs:annotation>
</xs:element>
<xs:element name="instance" abstract="true">
  <xs:annotation>
    <xs:documentation>renderable subject data</xs:documentation>
  </xs:annotation>
</xs:element>
<xs:element name="description" abstract="true">
  <xs:annotation>
    <xs:documentation>human-oriented description of reference subject</xs:documentation>
  </xs:annotation>
</xs:element>
<xs:element name="participation" abstract="true">
  <xs:annotation>
    <xs:documentation>value-chain atom</xs:documentation>
  </xs:annotation>
</xs:element>
<xs:element name="offerPart" abstract="true">
  <xs:annotation>
    <xs:documentation>offer of content</xs:documentation>
  </xs:annotation>
</xs:element>
<xs:element name="targetPart" abstract="true">
  <xs:annotation>
    <xs:documentation> target description</xs:documentation>
  </xs:annotation>
</xs:element>
<xs:element name="renderingSpecifier" abstract="true">
  <xs:annotation>
    <xs:documentation>abstract rendering hints</xs:documentation>
  </xs:annotation>
</xs:element>
<!--Utility Elements-->
<xs:element name="itemref">
  <xs:annotation>
```

```
<xs:documentation>reference to library element</xs:documentation>
</xs:annotation>
<xs:complexType>
  <xs:attribute name="idref" type="xs:IDREF" use="required"/>
</xs:complexType>
</xs:element>
<xs:element name="opaqueElement" abstract="true">
  <xs:annotation>
    <xs:documentation>generally uninterpretable element</xs:documentation>
  </xs:annotation>
</xs:element>
<xs:element name="certification" abstract="true">
  <xs:annotation>
    <xs:documentation>attributed claim of fact</xs:documentation>
  </xs:annotation>
</xs:element>
<!--Library Elements-->
<xs:element name="authorityLibrary">
  <xs:annotation>
    <xs:documentation>non-correlated collection of authority elements</xs:documentation>
  </xs:annotation>
<xs:complexType>
  <xs:sequence>
    <xs:element name="item" maxOccurs="unbounded">
      <xs:complexType>
        <xs:choice>
          <xs:element ref="authority"/>
          <xs:element ref="opaqueElement"/>
        </xs:choice>
        <xs:attribute name="id" type="xs:ID" use="required"/>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
</xs:complexType>
</xs:element>
<xs:element name="participationLibrary">
  <xs:annotation>
    <xs:documentation>non-correlated collection of participation elemnts</xs:documentation>
  </xs:annotation>
<xs:complexType>
  <xs:sequence>
    <xs:element name="item" maxOccurs="unbounded">
      <xs:complexType>
        <xs:choice>
          <xs:element ref="participation"/>
          <xs:element ref="opaqueElement"/>
        </xs:choice>
        <xs:attribute name="id" type="xs:ID" use="required"/>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
</xs:complexType>
</xs:element>
<!--Simple Concrete Substitutions-->
<xs:element name="server" substitutionGroup="authority">
  <xs:annotation>
    <xs:documentation>simple authority URL</xs:documentation>
  </xs:annotation>
<xs:complexType>
  <xs:simpleContent>
    <xs:extension base="xs:anyURI">
      <xs:attributeGroup ref="SCHEME"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
</xs:element>
<xs:element name="identifier" substitutionGroup="identification">
  <xs:annotation>
    <xs:documentation>simple content identifier</xs:documentation>
  </xs:annotation>
<xs:complexType>
  <xs:simpleContent>
    <xs:extension base="identifierType"/>
  </xs:simpleContent>
</xs:complexType>
</xs:element>
<xs:element name="location" substitutionGroup="instance">
  <xs:annotation>
    <xs:documentation>generic content locator</xs:documentation>
  </xs:annotation>
<xs:complexType>
  <xs:simpleContent>
    <xs:extension base="xs:anyURI">
      <xs:attributeGroup ref="SCHEME"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
</xs:element>
<xs:element name="bObject" substitutionGroup="instance">
  <xs:annotation>
    <xs:documentation>generic content data</xs:documentation>
  </xs:annotation>
<xs:complexType>
  <xs:simpleContent>
```

```
<xs:extension base="xs:base64Binary">
  <xs:attributeGroup ref="SCHEME"/>
</xs:extension>
</xs:simpleContent>
</xs:complexType>
</xs:element>
<xs:element name="consumer" substitutionGroup="targetPart">
  <xs:annotation>
    <xs:documentation>consumption environment identifier</xs:documentation>
  </xs:annotation>
<xs:complexType>
  <xs:simpleContent>
    <xs:extension base="xs:normalizedString">
      <xs:attributeGroup ref="SCHEME"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
</xs:element>
<xs:element name="proposition" substitutionGroup="offerPart">
  <xs:annotation>
    <xs:documentation>simple relational offer for outright purchase</xs:documentation>
  </xs:annotation>
<xs:complexType>
  <xs:sequence>
    <xs:element ref="identification"/>
    <xs:element name="packaging">
      <xs:annotation>
        <xs:documentation>package type identifier</xs:documentation>
      </xs:annotation>
<xs:complexType>
  <xs:simpleContent>
    <xs:extension base="xs:normalizedString">
      <xs:attributeGroup ref="SCHEME"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
</xs:element>
<xs:element name="price">
  <xs:annotation>
    <xs:documentation>execution and usage terms</xs:documentation>
  </xs:annotation>
<xs:complexType>
  <xs:simpleContent>
    <xs:extension base="xs:integer">
      <xs:attributeGroup ref="SCHEME"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
<xs:element name="contribution" substitutionGroup="participation">
  <xs:annotation>
    <xs:documentation>simple relational value-chain participation expression</xs:documentation>
  </xs:annotation>
<xs:complexType>
  <xs:sequence>
    <xs:element name="participant">
      <xs:complexType>
        <xs:simpleContent>
          <xs:extension base="xs:normalizedString">
            <xs:attributeGroup ref="SCHEME"/>
          </xs:extension>
        </xs:simpleContent>
      </xs:complexType>
    </xs:element>
    <xs:element name="role">
      <xs:complexType>
        <xs:simpleContent>
          <xs:extension base="xs:normalizedString">
            <xs:attributeGroup ref="SCHEME"/>
          </xs:extension>
        </xs:simpleContent>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
</xs:complexType>
</xs:element>
<xs:element name="title" substitutionGroup="description">
  <xs:annotation>
    <xs:documentation>descriptive information consisting of subject name only</xs:documentation>
  </xs:annotation>
<xs:complexType>
  <xs:simpleContent>
    <xs:extension base="xs:normalizedString">
      <xs:attribute name="language" type="xs:language"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
</xs:element>
<xs:element name="hidden" substitutionGroup="opaqueElement">
  <xs:annotation>
```

```
<xs:documentation>generic opaque element</xs:documentation>
</xs:annotation>
<xs:complexType>
  <xs:simpleContent>
    <xs:extension base="xs:base64Binary">
      <xs:attributeGroup ref="SCHEME"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
</xs:element>
</xs:schema>
```

Appendix B - References

1. **[XML]** Extensible Markup Language (XML) 1.0 (Second Edition); W3C Recommendation 6 October 2000; Tim Bray, Jean Paoli, C. M. Sperberg-McQueen, Eve Maler.
2. **[Schema Struct]** XML Schema Part 1: Structures; W3C Recommendation 2 May 2001; Henry S. Thompson, David Beech, Murray Maloney, Noah Mendelsohn
3. **[Schema Types]** XML Schema Part 2: Data Types; W3C Recommendation 2 May 2001; Paul V. Biron, Ashok Malhotra
4. **[XML Names]** Namespaces in XML; W3C Recommendation 14 January 1999; Tim Bray, Dave Hollander, Andrew Layman